Available online at www.sciencedirect.com
ScienceDirect
Theoretical Computer Science 377 (2007) 25–34
Theoretical Computer Science
www.elsevier.com/locate/tcs

Algorithms for long paths in graphs

Zhao Zhang[a,*,1], Hao Li[b,c,2]

[a] College of Mathematics and System Sciences, Xinjiang University, Urumqi, Xinjiang, 830046, China
[b] Laboratoire de Recherche en Informatique, UMR 8623, C.N.R.S.-Université de Paris-sud, 91405-Orsay cedex, France
[c] School of Mathematics and Statistics, Lanzhou University, 730000 Lanzhou, Gansu, China

Received 15 March 2006; received in revised form 11 October 2006; accepted 4 February 2007

Communicated by E. Pergola

Abstract

We obtain a polynomial algorithm in $O(nm)$ time to find a long path in any graph with n vertices and m edges. The length of the path is bounded by a parameter defined on neighborhood condition of any three independent vertices of the path. An example is given to show that this bound is better than several classic results.

Keywords: Hamiltonian; Long path; Neighborhood condition

1. Introduction and notation

It is a classic problem to find a long path or cycle in a graph. Since finding a Hamiltonian path/cycle in graphs is NP-hard, we are interested in finding a path with large length.

All graphs considered in this paper are undirected and simple. We follow the notation and terminology in [7]. For a graph $G = (V(G), E(G))$ and a subgraph H of G, the neighborhood of a vertex u in H is $N_H(u) = \{v \in V(H) : uv \in E(G)\}$. The degree of u in H is $d_H(u) = |N_H(u)|$. In the case $H = G$, we use $N(u)$ and $d(u)$ instead of $N_G(u)$ and $d_G(u)$. For simplicity, the graph itself is used to denote its set of vertices.

For a path $P = u_1u_2 \dots u_p$ and two indices $i < j$, denote by $P[u_i, u_j] = u_iu_{i+1} \dots u_j$, and $\overline{P}[u_j, u_i] = u_ju_{j-1} \dots u_i$. Define $P(u_i, u_j] = P[u_{i+1}, u_j]$, $P[u_i, u_j) = P[u_i, u_{j-1}]$ and $P(u_i, u_j) = P[u_{i+1}, u_{j-1}]$. For any i, $u_i^+ = u_{i+1}$ and $u_i^- = u_{i-1}$. For $A \subseteq P$, $A^+ = \{v^+ | v \in A\}$, $A^- = \{v^- | v \in A\}$.

We use $|P|$ to denote the number of vertices in a path P. Denote by $\sigma_2(G) = \min\{d(u) + d(v) : uv \notin E(G)\}$ and $\overline{\sigma}_3(G) = \min\{\sum_{i=1}^3 d(u_i) - \left|\bigcap_{i=1}^3 N(u_i)\right| : \{u_1, u_2, u_3\}$ is an independent set of $G\}$.

A Hamiltonian cycle (path, resp.) is a cycle (path, resp.) containing all vertices of the graph. A graph G is Hamiltonian if it has a Hamiltonian cycle. For an integer k, a graph is called k-connected if any two vertices cannot be separated by deleting less than k vertices in the graph.

* Corresponding author. Tel.: +86 13669969821.
E-mail addresses: zhzhao@xju.edu.cn (Z. Zhang), li@lri.fr (H. Li).
[1] The research is supported by NSFC (60603003) and XJEDU.
[2] The research is supported by NNSF of China (60373012).

0304-3975/$ - see front matter © 2007 Elsevier B.V. All rights reserved.
doi:10.1016/j.tcs.2007.02.012

We begin with the following basic results in Hamiltonian graph theory, which are due to Dirac, Ore and Flandrin, Jung and Li, respectively.

Theorem 1 ([8]). *Let G be a graph on $n \geq 3$ vertices. If the minimum degree δ is at least $n/2$, then G is Hamiltonian.*

Theorem 2 ([15]). *Let G be a graph on $n \geq 3$ vertices. If $\sigma_2(G) \geq n$, then G is Hamiltonian.*

Theorem 3 ([10]). *If G is a 2-connected graph of order n such that $\overline{\sigma}_3(G) \geq n$, then G is Hamiltonian.*

These results are generalized to circumferences of the graphs. The circumference $c(G)$ is the length of a longest cycle in G.

Theorem 4 ([8]). *If G is a 2-connected graph on $n \geq 3$ vertices, then $c(G) \geq \min\{n, 2\delta\}$.*

Theorem 5 ([4]). *Let G be a 2-connected graph on $n \geq 3$ vertices. Then $c(G) \geq \min\{n, \sigma_2(G)\}$.*

Theorem 6 ([17]). *Let G be a 3-connected graph on n vertices. Then $c(G) \geq \min\{n, \overline{\sigma}_3(G)\}$.*

As a consequence of Theorem 6, we have the following

Corollary 1. *Let G be a 2-connected graph on n vertices. Then there exists a path of at least $\min\{n, \overline{\sigma}_3(G) + 1\}$ vertices.*

Proof. Let D be a graph obtained from G by adding a new vertex w which is adjacent to every vertex of G. Then D is 3-connected. By Theorem 6, $c(D) \geq \min\{n, \overline{\sigma}_3(D)\}$. Since $\overline{\sigma}_3(D) \geq \overline{\sigma}_3(G) + 2$, we see that G has a path of at least $c(D) - 1 \geq \min\{n, \overline{\sigma}_3(G) + 1\}$ vertices. □

Since $\overline{\sigma}_3(G) \geq \sigma_2(G) \geq 2\delta$, we have the following two results:

Corollary 2. *Let G be a 2-connected graph on n vertices. Then there exists a path of at least $\min\{n, \sigma_2(G) + 1\}$ vertices.*

Corollary 3. *Let G be a 2-connected graph on n vertices. Then there exists a path of at least $\min\{n, 2\delta + 1\}$ vertices.*

In this paper, we will generalize the above corollaries by giving a new lower bound for the length of a longest path, using a neighborhood condition of three independent vertices, one of which is an end of the path!

Since the problem of deciding whether a graph has a Hamiltonian path is NP-complete, it is interesting to find a long path in a network which can be realized by a polynomial algorithm.

The first algorithms for finding long paths are due to Monien [14] and Bodlaender [6], both finding paths of length $O(\log L / \log\log L)$, where L is the length of a longest path. By introducing the important method of *color-coding*, Alon, Yuster and Zwick [2] presented a randomized algorithm, which they derandomized, that finds paths of length $O(\log L)$ with performance ratio $O(|V|/\log|V|)$. When restricted to Hamiltonian graphs, the '$\log|V|$-barrier' was broken by Vishwanathan [16] whose result was later generalized to general graphs by Björklund and Husfeldt [5] for finding a path of length $O((\log L/\log\log L)^2)$ with performance ratio $O(|V|(\log\log|V|)^2/\log^2|V|)$. In sparse Hamiltonian graphs, Feder, Motwani and Subi [9] find even longer paths.

The hardness results, proved by Karger, Motwani, Ramkumar [13], and later Bazgan, Santha and Tuza [3] show that the longest path problem is not constant approximable even in cubic Hamiltonian graphs. Furthermore, for any $\epsilon > 0$, this problem cannot be approximated within $2^{O(\log^{1-\epsilon}|V|)}$ unless NP⊆DTIME($2^{O(\log^{1/\epsilon}|V|)}$).

In [11], Kocay and Li described an algorithm finding a long path containing a selected vertex. In this paper, we give an algorithm with time complexity $O(nm)$, by which we can find a long path with a length related to an end vertex of the path.

Some notation will be used in this paper. For a subgraph H and three vertices x, y, z, denote by

$$\Gamma_H(x, y, z) = d_H(x) + d_H(y) + d_H(z) - |N_H(x) \cap N_H(y) \cap N_H(z)|.$$

For $x \in H$, denote by

$$\Gamma_3(x, H) = \min\{\Gamma_H(x, y, z) \mid : y, z \in H \text{ and } x, y, z \text{ are independent}\}.$$

Clearly $\Gamma_3(x, H) \geq \overline{\sigma}_3(G)$.

The main result is the following:

Theorem 7. *Let G be a 2-connected graph of order $n \geq 3$. Then there exists a vertex x and a path P such that x is one end vertex of P and P contains at least* $\min\{n, \Gamma_3(x, P) + 1\}$ *vertices. Furthermore, P can be found in $O(nm)$ time.*

Theorem 7 is best possible in the following sense. Suppose d, f, r are three integers with $d \geq 8$, $3 \leq f \leq d-5$, and $r \geq 2$. Let G be the graph obtained from d disjoint graphs G_i $(1 \leq i \leq d)$ with $G_i \cong K_r$ $(1 \leq i \leq f)$ and $G_j \cong K_1$ $(f+1 \leq i \leq d)$, by adding edges from G_{d-1} and G_d to all the other vertices. It is easy to see that there is a path P containing all the vertices in G_i $(i = 1, 2, 3, d-1, d)$ with two end vertices $x_1 \in G_1$ and $x_2 \in G_2$ respectively. Clearly, P is a longest path with $3r+2 = d(x_1) + d(x_2) + d(x_3) - |N(x_1) \cap N(x_2) \cap N(x_3)| + 1$ vertices, where x_3 is a vertex in G_3. So the bound in Theorem 7 is sharp. Furthermore, the same example shows that our result is better than the corollaries since $\overline{\sigma}_3(G) = 4 < |P| + 1$.

2. Proof of the main theorem

The idea of our proof of Theorem 7 is as follows. Let $P_1 = u_0u_1 \dots u_p$ be a maximal path (in the sense of inclusion of vertices), and $P_2 = v_0v_1 \dots v_q$ with

(a) $P_1 \cap P_2 = \{v_0\} = \{u_c\}$,
(b) subject to (a), c is as large as possible, and
(c) subject to (a) and (b), q is as large as possible.

Then a cycle P_V called vine of P_1 (which will be defined later) is found. Based on P_1, P_2 and P_V, a path P is constructed such that

$$v_q, u_p, u_0 \text{ are three independent vertices on } P \text{ with} \tag{1}$$
$$v_q \text{ or } u_0 \text{ being one end of } P, \text{ and} \tag{2}$$
$$N(v_q) \cup N(u_p) \cup N(u_0) \subseteq P, \tag{3}$$
$$\Gamma_P(v_q, u_p, u_0) \leq |P| - 1. \tag{4}$$

With these properties, it is easy to see that P is a path with the desired length.

From an algorithmic point of view, finding a maximal path P_1 requires a lot of work. However, to ensure that the path P we find has the desired length in Theorem 7, we do not need all properties of a maximal path. In fact, properties (1)–(4) are essential for our purpose, and to ensure that P satisfies properties (1)–(4), only nine operations to extend P_1 are sufficient, which are introduced in the following.

Circumstance 1: There is a vertex $v \in V(G) \setminus V(P_1)$ which is adjacent to one end of P_1.
Operation 1: Extend P_1 by adding v.

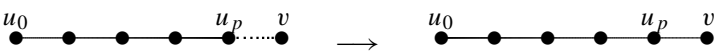

Fig. 1. Extending P_1 by Operation 1.

Circumstance 2: There is a vertex $v \in V(G) \setminus V(P_1)$ such that $u_i \in N_{P_1}(v)$ and u_{i+1} is connected to v by a path internally disjoint from P_1.
Operation 2: Reset $P_1 = u_0u_1 \dots u_iv \dots u_{i+1} \dots u_p$.

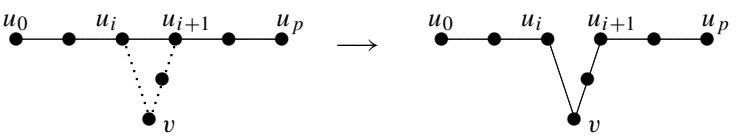

Fig. 2. Extending P_1 by Operation 2.

Circumstance 3: u_0 is adjacent to u_p, and $V(G) \setminus V(P_1) \neq \emptyset$.

Operation 3: Let v be a vertex in $V(G) \setminus V(P_1)$ which is adjacent to some vertex u_i on P_1. Reset $P_1 = vu_iu_{i-1}\ldots u_0u_pu_{p-1}\ldots u_{i+1}$.

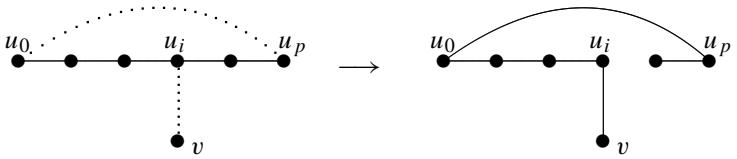

Fig. 3. Extending P_1 by Operation 3.

Circumstance 4: $u_i \in N_{P_1}(u_0) \cap N_{P_1}(u_p)^+ \neq \emptyset$ and $V(G) \setminus V(P_1) \neq \emptyset$.

Operation 4: Reset $P_1 = u_{i-1}u_{i-2}\ldots u_0u_iu_{i+1}\ldots u_p$, and then extend it further by Operation 3.

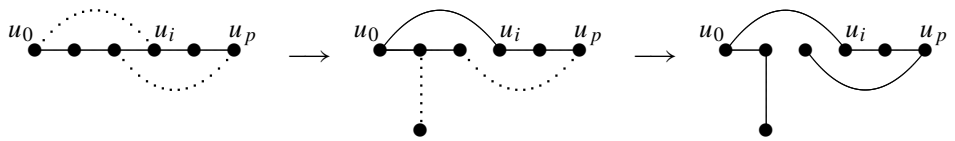

Fig. 4. Extending P_1 by Operation 4.

Circumstance 5: There is a vertex $u_i \in N(u_p)$ with u_{i+1} having some neighbor v outside of P_1.

Operation 5: Reset $P_1 = u_0u_1\ldots u_iu_pu_{p-1}\ldots u_{i+1}v$.

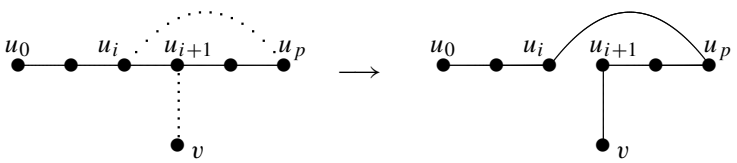

Fig. 5. Extending P_1 by Operation 5.

Circumstance 6: There is a vertex $u_i \in N(u_0)$ with u_{i-1} having some neighbor v outside of P_1.

Operation 6: Reset $P_1 = vu_{i-1}u_{i-2}\ldots u_0u_iu_{i+1}\ldots u_p$.

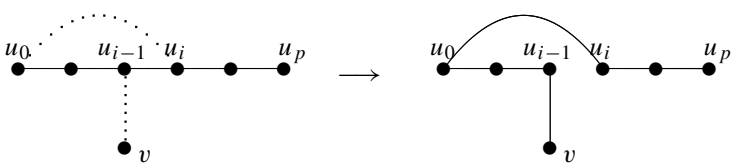

Fig. 6. Extending P_1 by Operation 6.

Circumstance 7: There is a vertex $u_i \in N(u_p)$ with u_{i-1} having some neighbor v outside of P_1, and there is an index $j > i$ such that $u_j \in N(u_0)$.

Operation 7: Reset $P_1 = vu_{i-1}u_{i-2}\ldots u_0u_ju_{j-1}\ldots u_iu_pu_{p-1}\ldots u_{j+1}$.

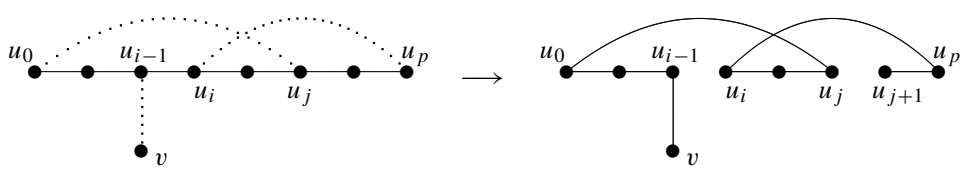

Fig. 7. Extending P_1 by Operation 7.

Circumstance 8: $u_i \in N_{P_1[u_1,u_c)}(u_p) \cap N_{P_1[u_1,u_c)}(v_q)^+ \neq \emptyset$.

Operation 8: Reset $P_1 = u_0u_1\ldots u_{i-1}v_qv_{q-1}\ldots v_1u_cu_{c+1}\ldots u_pu_iu_{i+1}\ldots u_{c-1}$.

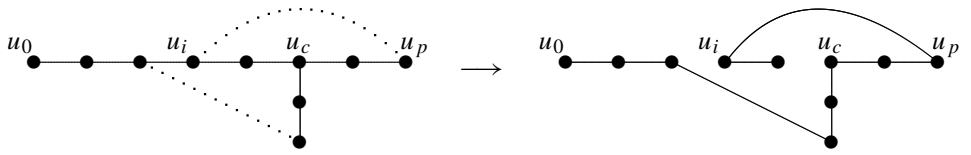

Fig. 8. Extending P_1 by Operation 8.

Circumstance 9: $u_i \in N_{P_1[u_1,u_c)}(u_p) \cap N_{P_1[u_1,u_c)}(u_p)^+ \neq \emptyset$.

Operation 9: Reset $P_1 = u_0u_1 \dots u_{i-1}u_pu_iu_{i+1} \dots u_{p-1}$.

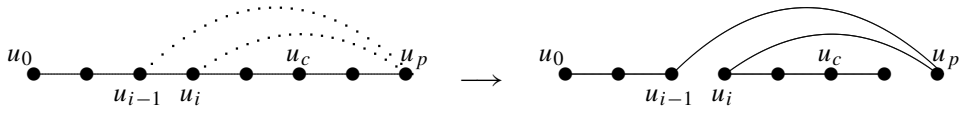

Fig. 9. Extending P_1 by Operation 9.

Note that except for Operation 9, all operations extend P_1 by at least one vertex. And Operation 9 increases c by one.

Some of the above operations are also used in dealing with Hamiltonian problems. For example, Circumstance 3 occurs in [1] for finding Hamiltonian cycles.

Algorithm 1.
Input: A connected graph G.
Output: Either a Hamiltonian path P_1, or two paths P_1 and P_2 sharing only one common vertex u_c, and P_1 cannot be extended by Operations 1 to 9.
Step 1. Set $P_1 = u_0$ where u_0 is an arbitrary vertex in G.
Step 2. Extend P_1 repeatedly by Operation 1 until such operation can no longer be carried out.
Step 3. If $V(G) \setminus V(P_1) = \emptyset$, then output P_1 which is a Hamiltonian path; stop. Else, if one of circumstances 2 to 7 happens, then extend P_1 by the corresponding operation; go to Step 2.
Step 4. If $V(G) \setminus V(P_1) = \emptyset$, then output P_1; stop. Else, let u_c be the last vertex on P_1 which has a neighbor outside of P_1; set $v_0 = u_c$; find a maximal path P_2 in $G - P_1$ starting at v_0, i.e., as long as there is a vertex $v \in V(G) - V(P_1 \cup P_2)$ adjacent to the other end of P_2, then extend P_2 by adding v.
Step 5. If circumstance 8 or circumstance 9 happens, then extend P_1 by the corresponding operation; go to Step 2. Else, output P_1, P_2 and u_c; stop. □

Given a path $P = u_0u_1 \dots u_p$, let $\mathcal{Q} := \{Q_\ell[u_{i_\ell}, u_{j_\ell}] : 1 \leq \ell \leq m\}$ be a set of internally disjoint paths such that $Q_\ell \cap P = \{u_{i_\ell}, u_{j_\ell}\}$ and

$$0 = i_1 < i_2 < j_1 \leq i_3 < j_2 \leq i_4 \cdots \leq i_m < j_{m-1} < j_m = p.$$

Denote by $\mathcal{P}$ the set of segments of P divided by u_{i_ℓ}'s and u_{j_ℓ}'s. A vine of P is composed of elements in $\mathcal{Q} \cup \mathcal{P}$ alternatively (see Fig. 10).

For our purpose, we will find a vine P_V of P in a 2-connected graph with $N_P(u_0) \cup N_P(u_p) \subseteq P_V$, which can be realized by the following algorithm.

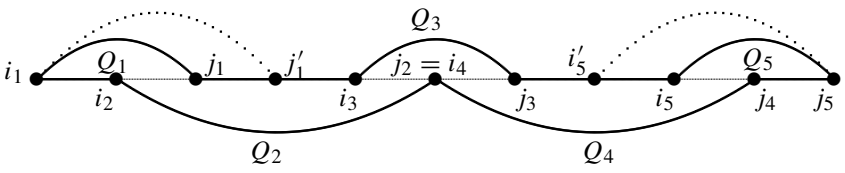

Fig. 10. The vine is indicated by the bold lines.

Algorithm 2.
Input: A path $P = u_0u_1 \dots u_p$.
Output: A vine P_V with $N(u_0) \cup N(u_p) \subseteq P_V$.
Step 1. Set $i_1 = 0$. Let j_1' be the largest index such that $u_{j_1'}$ is adjacent to u_0. Set $\ell = 2$, $v = u_{j_1'}$, $w = u_0$.
Step 2. Find a path Q_ℓ in $G - v$ internally disjoint with P, connecting a vertex $u_{i_\ell} \in P[w, v^-]$ with a vertex $u_{j_\ell} \in P[v^+, u_p]$, such that j_ℓ is as large as possible (such a path always exists since G is 2-connected).
Step 3. If $j_\ell = p$, then choose i_ℓ as large as possible, go to Step 4. Else, set $w = v$, $v = u_{j_\ell}$, $\ell = \ell + 1$, go to Step 2.
Step 4. Set j_1 to be the first index in the segment $[u_{i_2}^+, u_{j_1'}]$ such that $u_{j_1} \in N_P(u_0)$.
Step 5. If ℓ is even, then let

$$P_V := \underline{Q_1}[u_{i_1}, u_{j_1})\underline{P}[u_{j_1}, u_{i_3})\underline{Q_3}[u_{i_3}, u_{j_3})P[u_{j_3}, u_{i_5}) \dots Q_{\ell-1}[u_{i_{\ell-1}}, u_{j_{\ell-1}})P[u_{j_{\ell-1}}, u_{j_\ell})$$
$$\overline{Q_\ell}[u_{j_\ell}, u_{i_\ell})\overline{P}[u_{i_\ell}, u_{j_{\ell-2}})\overline{Q_{\ell-2}}[u_{j_{\ell-2}}, u_{i_{\ell-2}})\overline{P}[u_{i_{\ell-2}}, u_{j_{\ell-4}}) \dots \overline{Q_2}[u_{j_2}, u_{i_2})\overline{P}[u_{i_2}, u_{i_1}],$$

and if ℓ is odd, then let

$$P_V := Q_1[u_{i_1}, u_{j_1})\underline{P}[u_{j_1}, u_{i_3})\underline{Q_3}[u_{i_3}, u_{j_3})P[u_{j_3}, u_{i_5}) \dots Q_{\ell-2}[u_{i_{\ell-2}}, u_{j_{\ell-2}})P[u_{j_{\ell-2}}, u_{i_\ell})$$
$$Q_\ell[u_{i_\ell}, u_{j_\ell})\overline{P}[u_{j_\ell}, u_{j_{\ell-1}})\overline{Q_{\ell-1}}[u_{j_{\ell-1}}, u_{i_{\ell-1}})\overline{P}[u_{i_{\ell-1}}, u_{j_{\ell-3}}) \dots \overline{Q_2}[u_{j_2}, u_{i_2})\overline{P}[u_{i_2}, u_{i_1}]. \quad \square$$

How Algorithm 2 works can be illustrated by Fig. 10 ($m = 5$ in this figure).

Suppose m is the ℓ-value at the end of the algorithm. Then $u_{j_m} = u_p$. By the choice of j_ℓ in Step 2, we see that $N_P(u_p) \subseteq P[u_{j_{m-2}}, u_{p-1}]$. By the choice of i_m in Step 3, we have $N_P(u_p) \cap P[u_{i_m}^+, u_{j_{m-1}}^-] = \emptyset$. So

$$N_P(u_p) \subseteq P[u_{j_{m-2}}, u_{p-1}] - P[u_{i_m}^+, u_{j_{m-1}}^-] \subseteq P_V. \tag{5}$$

Similarly, by the choice of j_1' in Step 1 and the choice of j_1 in Step 4, we have

$$N_P(u_0) \subseteq P[u_1, u_{i_3}] - P[u_{i_2}^+, u_{j_1}^-] \subseteq P_V. \tag{6}$$

The next algorithm finds a path P satisfying conditions (1)–(4). For simplicity, we abuse the notation a little by, for example, using $P_V(u_{i_\ell}, u_c]$ to denote $P_V(u_{i_\ell}, u_{j_{\ell-1}}]\,\overline{P}_1(u_{j_{\ell-1}}, u_c]$ when $u_c \in (u_{i_\ell}, u_{j_{\ell-1}})$. The same denotation is used in the remaining of this paper when there is no danger of confusion.

Algorithm 3.
Input: A 2-connected graph G.
Output: A vertex x and a path P with length at least $\min\{|G|, \Gamma_3(x, P) + 1\}$ such that x is one end vertex of P.
Step 1. Use Algorithm 1 to find P_1, P_2 and u_c. If P_1 is Hamiltonian, then set $P = P_1$; stop.
Step 2. Use Algorithm 2 to find P_V.
Step 3. Let ℓ be the largest integer such that $u_c \in (u_{i_\ell}, u_{j_\ell})$. If $(u_{i_\ell}, u_c) \cap N(v_q) = \emptyset$, then set $u_g = u_c$, $tag = 0$. Else, let u_g be the first vertex in $(u_{i_\ell}, u_c) \cap N(v_q)$, set $tag = 1$.
Step 4. If $\ell = 1$, then set $x = v_q$ and $P = \overline{P_1}[u_{c-1}, u_0]P_V(u_0, u_c]P_2(v_0, v_q]$ (see Fig. 11(a)), stop.
Step 5. If $(u_{j_{\ell-1}}, u_g) \cap N(u_0) \neq \emptyset$, then set $x = v_q$ and $P = \overline{P_1}[u_{c-1}, u_{j_{\ell-1}}]P_1[u_0, u_{i_\ell}]P_V(u_{i_\ell}, u_c]P_2(v_0, v_q]$ (see Fig. 11(b)), stop.
Step 6. Set $x = u_0$. If $[u_{j_{\ell-1}}, u_g) \cap N(u_p) = \emptyset$, then set $P = P_1[u_0, u_{i_\ell}]P_V(u_{i_\ell}, u_c]$ (see Fig. 11(c) or (d)). Else, let u_f be the last vertex in $[u_{j_{\ell-1}}, u_g) \cap N(u_p)$ and set $P = P_1[u_0, u_f]\overline{P_1}[u_p, u_c]$ (see Fig. 11(e)).
Step 7. If $tag = 0$, then set $P = PP_2(v_0, v_q]$. Else, set $P = P\overline{P_1}(u_c, u_g]\overline{P_2}[v_q, v_1]$.

We will show that the path P found by Algorithm 3 indeed satisfies conditions (1)–(4). For this purpose, we need the following lemmas.

Lemma 1. *Let $P = u_0u_1 \dots u_p$ be a path in G and $y, z \in V(G) - P$ such that $N_P(z) \cap N_P(y)^+ = \emptyset$. Then*

$$d_P(y) + d_P(z) \leq |P| + 1. \tag{7}$$

The equality holds only if $u_p \in N_P(y)$. Furthermore, if $N_P(y) \cap N_P(y)^+ = \emptyset$, then equality holds only when $u_p \in N_P(y) \cap N_P(z)$.

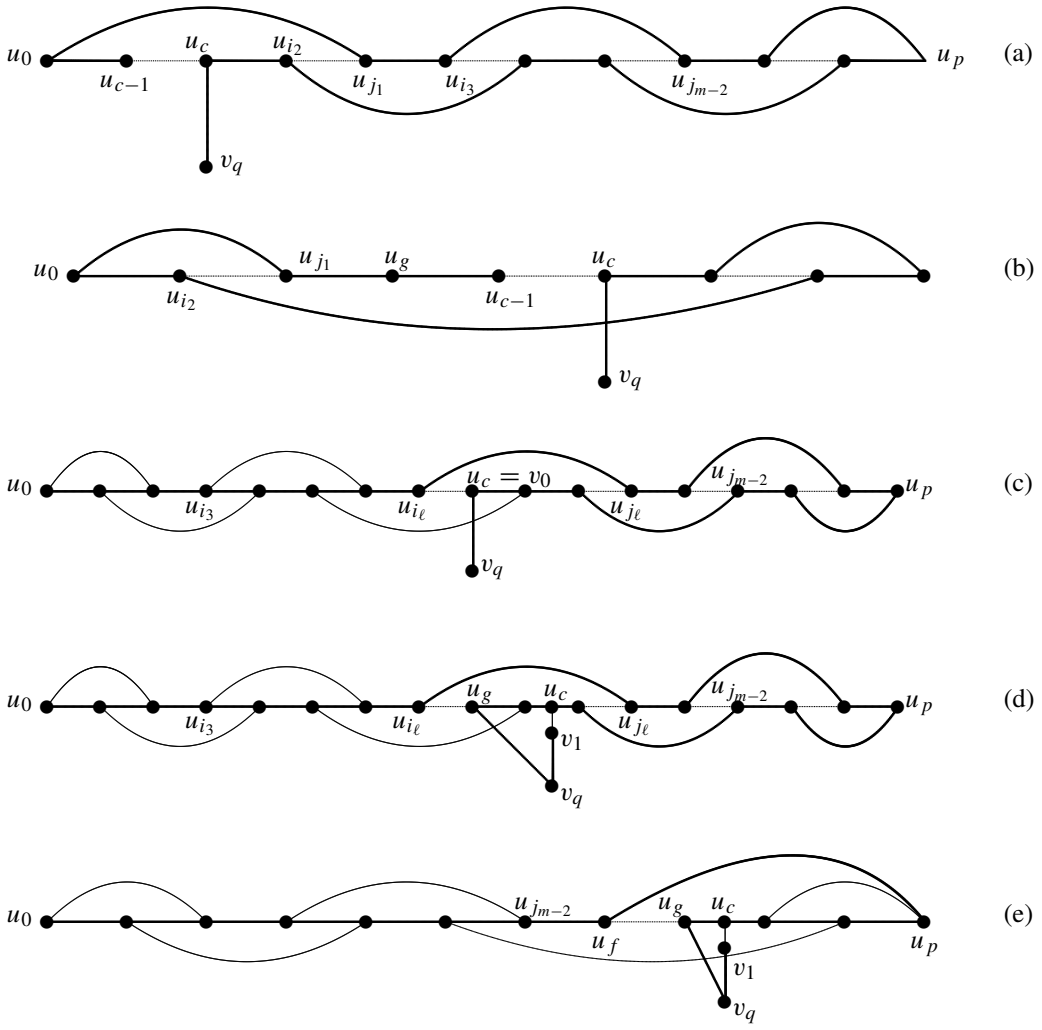

Fig. 11. Path P is indicated by bold lines.

Proof. Since $(N_P(z) \cup (N_P(y) - \{u_p\})^+) \subseteq V(P)$ and $N_P(z) \cap N_P(y)^+ = \emptyset$, we have $|P| \geq |N_P(z)| + |(N_P(y) - \{u_p\})^+| \geq d_P(z) + d_P(y) - 1$. Equality holds only if

$$V(P) = N_P(z) \cup (N_P(y) - \{u_p\})^+ \tag{8}$$

and $u_p \in N_P(y)$. Furthermore, if $N_P(y) \cap N_P(y)^+ = \emptyset$ and equality holds, then it follows from $u_p \in N_P(y)$ that $u_p \notin N_P(y)^+$. By (8), we have $u_p \in N_P(z)$. □

Lemma 2. *Let $P = u_0u_1 \ldots u_p$ be a path in G and $x, y, z \in V(G) - P$ such that $N_P(x) \cap N_P(x)^+ = (N_P(y) \cup N_P(z)) \cap N_P(x)^+ = N_P(y) \cap N_P(y)^+ = N_P(z) \cap N_P(y)^+ = \emptyset$. Then*

$$\Gamma_P(x, y, z) \leq |P| + 1. \tag{9}$$

Furthermore, if equality holds and $u_p \notin N_P(x)$, then $u_p \in N_P(y) \cap N_P(z)$.

Proof. If $N_P(x) = \emptyset$, then it follows from Lemma 1 that

$$\Gamma_P(x, y, z) = d_P(y) + d_P(z) \leq |P| + 1, \tag{10}$$

with equality only when $u_p \in N_P(y) \cap N_P(z)$.

So, suppose $N_P(x) = \{u_{i_1}, u_{i_2}, \ldots, u_{i_t}\} \neq \emptyset$. Consider a segment $P(u_{i_j}, u_{i_{j+1}}]$, $1 \leq j < t$. By Lemma 1, noting that $u_{i_j+1} \notin N_P(y) \cup N_P(z)$, we see that

$$d_{P(u_{i_j},u_{i_{j+1}}]}(y) + d_{P(u_{i_j},u_{i_{j+1}}]}(z)) \leq |P(u_{i_j}, u_{i_{j+1}}]|, \tag{11}$$

with equality only when $u_{i_{j+1}} \in N(y) \cap N(z)$. Therefore

$$\begin{aligned} \Gamma_{P(u_{i_j},u_{i_{j+1}}]}(x, y, z) &= 1 + d_{P(u_{i_j},u_{i_{j+1}}]}(y) + d_{P(u_{i_j},u_{i_{j+1}}]}(z) - |\{u_{i_{j+1}}\} \cap N(y) \cap N(z)| \\ &\leq |P(u_{i_j}, u_{i_{j+1}}]|. \end{aligned} \tag{12}$$

For the first segment $P[u_0, u_{i_1}]$ and the last segment $P(u_{i_t}, u_p]$, similar to the above we may get

$$\Gamma_{P[u_0,u_{i_1}]}(x, y, z) \leq |P[u_0, u_{i_1}]| + 1 \tag{13}$$

and

$$\Gamma_{P(u_{i_t},u_p]}(x, y, z) \leq |P(u_{i_t}, u_p]|. \tag{14}$$

Then (9) follows by adding (12)–(14) together. If equality holds for (9), then equality also holds for (14). If furthermore $u_p \notin N_P(x)$, then similar to the deduction of (11), we have

$$\Gamma_{P(u_{i_t},u_p]}(x, y, z) = d_{P(u_{i_t},u_p]}(y) + d_{P(u_{i_t},u_p]}(z) \leq |P(u_{i_t}, u_p]|,$$

with equality only when $u_p \in N_P(y) \cap N_P(z)$. □

Next, we will prove the main theorem.

Proof of Theorem 7. Since each of the nine operations either extends P_1 by at least one vertex or increases c by one, at most $O(n)$ extensions are needed. Furthermore, each extension can be completed in $O(m)$ time by graph searching (see for example [12]). For the same reason, the time complexity of Algorithms 2 and 3 is also $O(m)$. So, P can be found in $O(nm)$ time. Next, we will prove that P satisfies conditions (1)–(4), and thus has the desired length.

Without loss of generality, we assume that G has no Hamiltonian path. Let $P_1 = u_0u_1 \ldots u_p$ and $P_2 = v_0v_1 \ldots v_q$ be the paths found by Algorithm 1, P_V the vine found by Algorithm 2, and m the ℓ-value at the end of Algorithm 2. By Operations 1 and 3, u_0, u_p, v_q are independent (Condition (1)). Condition (2) is obviously satisfied by the definition of the path P in Algorithm 3. Furthermore,

$$N_{P_1}(v_q) \cap N_{P_1}(v_q)^+ = \emptyset \text{ (by Operation 2)}, \tag{15}$$

$$N_{P_1[u_1,u_c)}(u_p) \cap N_{P_1[u_1,u_c)}(v_q)^+ = \emptyset \text{ (by Operation 8)}, \tag{16}$$

$$N_{P_1[u_1,u_c)}(u_0) \cap N_{P_1[u_1,u_c)}(v_q)^+ = \emptyset \text{ (by Operation 6)}, \tag{17}$$

$$N_{P_1[u_1,u_c)}(u_p) \cap N_{P_1[u_1,u_c)}(u_p)^+ = \emptyset \text{ (by Operation 9)}, \tag{18}$$

$$N_{P_1}(u_0) \cap N_{P_1}(u_p)^+ = \emptyset \text{ (by Operation 4)}. \tag{19}$$

The above conditions hold since otherwise P_1 and P_2 can be extended by the operations indicated in the brackets (see Figs. 1–9).

Note that neither u_0 nor u_p can have neighbors outside of P_1 since P_1 can no longer be extended by Operation 1. So, $N(u_0) = N_{P_1}(u_0)$, $N(u_p) = N_{P_1}(u_p)$, and thus it follows from (5) and (6) that (see Fig. 10 for illustration)

$$N(u_0) \subseteq P_1[u_1, u_{i_3}] - P_1[u_{i_2}^+, u_{j_1}^-] = P_1(u_0, u_{i_2}] \cup P_1[u_{j_1}, u_{i_3}], \tag{20}$$

$$N(u_p) \subseteq P_1[u_{j_{m-2}}, u_{p-1}] - P_1[u_{i_m}^+, u_{j_{m-1}}^-] = P_1[u_{j_{m-2}}, u_{i_m}] \cup P_1[u_{j_{m-1}}, u_p). \tag{21}$$

By the definition in Algorithm 1,

$$N(v_q) \subseteq P_1[u_1, u_c] \cup P_2. \tag{22}$$

Recall that ℓ is such that $u_c \in P_1(u_{i_\ell}, u_{j_\ell})$. It follows from (22) that the only possible neighbors of v_q which may be missed lie in the segment (u_{i_ℓ}, u_c). However, this can be compensated by the choice of u_g (Step 3 and Step 7 of Algorithm 3). So, $N(v_q) \subseteq P$. If $\ell \geq 3$, then $N(u_0) \subseteq P$ by (20). If $\ell \leq 2$, then by noting that $[u_g, u_c] \subseteq P$ (Step 7), we also have $N(u_0) \subseteq P$ by the definition of P in Step 4 and Step 5. Similarly, u_f is taken to ensure that $N(u_p) \subseteq P$

(Step 6). So, Condition (3) is satisfied. In the following, we will show Condition (4). To this end, we first prove the following three claims.

Claim 1. *Suppose* $Q = u_i u_{i+1} \dots u_{c-1}$ $(i > 0)$. *Then* $\Gamma_Q(v_q, u_p, u_0) \le |Q|$.

By taking $x = v_q$, $y = u_p$, $z = u_0$ in Lemma 2, and by (1) and (15)–(19), we see that

$$\Gamma_Q(v_q, u_p, u_0) \le |Q| + 1. \tag{23}$$

Note that $u_{c-1} \notin N(v_q)$ since otherwise P_1 can be extended by Operation 2. If equality holds in (23), then $u_{c-1} \in N(u_0) \cap N(u_p)$ by Lemma 2, and thus P_1 can be extended by Operation 5, a contradiction.

Claim 2. $\Gamma_{P_V[u_{j_\ell}, u_c]}(v_q, u_p, u_0) \le |P_V[u_{j_\ell}, u_c]|$ *when* $\ell \ge 2$ *and* $\Gamma_{P_V[u_{j_1}, u_c]}(v_q, u_p, u_0) \le |P_V[u_{j_1}, u_c]| + 1$ *when* $\ell = 1$.

Note that condition (19) is equivalent to $N_{\overline{P}_1}(u_p) \cap U_{\overline{P}_1}(u_0)^+ = \emptyset$. By taking $P = \overline{P}_1[u_{i_{\ell+1}}, u_{c+1}]$ and $y = u_0$, $z = u_p$ in Lemma 1, we have

$$d_{\overline{P}_1[u_{i_{\ell+1}} u_{c+1}]}(u_0) + d_{\overline{P}_1[u_{i_{\ell+1}} u_{c+1}]}(u_p) \le |\overline{P}_1[u_{i_{\ell+1}} u_{c+1}]| + 1,$$

with equality only when $u_{c+1} \in N(u_0)$. Since P_1 cannot be extended by Operation 6, we see that $u_{c+1} \notin N(u_0)$, and thus

$$d_{\overline{P}_1[u_{i_{\ell+1}} u_{c+1}]}(u_0) + d_{\overline{P}_1[u_{i_{\ell+1}} u_{c+1}]}(u_p) \le |\overline{P}_1[u_{i_{\ell+1}} u_{c+1}]|,$$

or equivalently,

$$d_{P_1(u_c, u_{i_{\ell+1}}]}(u_0) + d_{P_1(u_c, u_{i_{\ell+1}}]}(u_p) \le |P_1(u_c, u_{i_{\ell+1}}]|.$$

By taking $P = P_1[u_{j_1}, u_{i_3}]$ and $y = u_p$, $z = u_0$ in Lemma 1, it follows from Condition (19) that

$$d_{P_1[u_{j_1}, u_{i_3}]}(u_0) + d_{P_1[u_{j_1}, u_{i_3}]}(u_p) \le |P_1[u_{j_1}, u_{i_3}]| + 1.$$

So, when $\ell = 1$ (see Fig. 11),

$$\begin{aligned}
&d_{P_V[u_{j_1}, u_c)}(u_0) + d_{P_V[u_{j_1}, u_c)}(u_p) \\
&\quad = d_{P_1(u_c, u_{i_2}]}(u_0) + d_{P_1(u_c, u_{i_2}]}(u_p) + d_{P_1[u_{j_1}, u_{i_3}]}(u_0) + d_{P_1[u_{j_1}, u_{i_3}]}(u_p) + d_{P_1[u_{j_2}, u_p) \cap P_V}(u_p) \\
&\quad \le |P_1(u_c, u_{i_2}]| + |P_1[u_{j_1}, u_{i_3}]| + 1 + |P_1[u_{j_2}, u_p) \cap P_V| \\
&\quad = |P_1(u_c, u_p) \cap P_V| + 1 = |P_1[u_c, u_p] \cap P_V| - 1 = |P_V[u_{j_1}, u_c]| - 1,
\end{aligned}$$

and when $\ell \ge 2$,

$$\begin{aligned}
d_{P_V[u_{j_\ell}, u_c)}(u_0) + d_{P_V[u_{j_\ell}, u_c)}(u_p) &= d_{P_1(u_c, u_{i_{\ell+1}}]}(u_0) + d_{P_1(u_c, u_{i_{\ell+1}}]}(u_p) + d_{P_1[u_{j_\ell}, u_p) \cap P_V}(u_p) \\
&\le |P_1(u_c, u_{i_{\ell+1}}]| + |P_1[u_{j_\ell}, u_p) \cap P_V| \\
&= |P_1(u_c, u_p) \cap P_V| = |P_1[u_c, u_p] \cap P_V| - 2 = |P_V[u_{j_\ell}, u_c]| - 2.
\end{aligned}$$

Since $N(v_q) \cap P_V[u_{j_\ell}, u_c) = \emptyset$, we have

$$\Gamma_{P_V[u_{j_\ell}, u_c]}(v_q, u_p, u_0) = d_{P_V[u_{j_\ell}, u_c)}(u_0) + d_{P_V[u_{j_\ell}, u_c)}(u_p) + \Gamma_{\{u_c\}}(v_q, u_p, u_0).$$

Then the claim follows by noting that $\Gamma_{\{u_c\}}(v_q, u_p, u_0) \le 2$.

Claim 3. *Suppose* $Q = u_0 u_1 \dots u_i$ $(i < c)$. *Then* $\Gamma_Q(v_q, u_p, u_0) \le |Q|$. *If furthermore* $i = c - 1$, *then* $\Gamma_Q(v_q, u_p, u_0) \le |Q| - 1$.

In fact, by Lemma 2 and Conditions (15)–(19), we have

$$\Gamma_Q(v_q, u_p, u_0) = \Gamma_{Q \setminus u_0}(v_q, u_p, u_0) \le |Q \setminus u_0| + 1 = |Q|.$$

If furthermore $i = c - 1$, then the above inequality becomes strict by Claim 1.

Clearly,

$$\Gamma_{P_2(v_0,v_q]}(v_q, u_p, u_0) = d_{P_2(v_0,v_q)}(v_q) \leq |P_2(v_0, v_q]| - 1. \tag{24}$$

The path P can be divided into sections such that each section falls into one category of Claims 1–3 or inequality (24) (see Fig. 11). Note that there is a '-1' in the right hand side of inequality (24), and at most one '$+1$' in the right hand side of the inequalities of Claim 2 (when $\ell = 1$). So, Condition (4) is satisfied for $\ell \geq 2$. When $\ell = 1$, we have

$$\begin{aligned}\Gamma_P(v_q, u_p, u_0) &= \Gamma_{\overline{P}_1[u_{c-1},u_0]}(v_q, u_p, u_0) + \Gamma_{P_V[u_{j_1},u_c]}(v_q, u_p, u_0) + \Gamma_{P_2(v_0,v_q]}(v_q, u_p, u_0)\\ &\leq (|\overline{P}_1[u_{c-1}, u_0]| - 1) + (|P_V[u_{j_1}, u_c]| + 1) + (|P_2(v_0, v_q]| - 1)\\ &= |P| - 1.\end{aligned}$$

Condition (4) is also satisfied. □

Acknowledgements

We would like to appreciate the referees for directing our attention to some results related to our aspect of study, and for pointing out the flaws in our manuscript, which helped us a lot in improving the clarity of this paper.

References

[1] M. Albertson, Finding Hamiltonian cycles in ore graphs, Congr. Numer. 58 (1987) 25–27.
[2] N. Alon, R. Yuster, U. Zwick, Color-coding: A new method for finding simple paths, cycles and other small subgraphs within large graphs, in: Proceedings of STOC, 1994, pp. 326–335.
[3] C. Bazgan, M. Santha, Z. Tuza, On the approximability of finding a(nother) Hamiltonian cycles in cubic Hamiltonian graphs, in: Proceedings of STACS, 1998.
[4] J.-C. Bermond, On Hamiltonian walks, in: Proc. Fifth British Combinatorial Conference, Aberdeen, 1975, Util. Math. (1975) 41–51.
[5] A. Björklund, T. Husfeldt, Finding a path of superlogarithmic length, SIAM J. Comput. 32 (2003) 1395–1402.
[6] H.L. Bodlaender, On linear time minor tests with depth-first search, J. Algorithms 14 (1993) 1–23.
[7] J.A. Bondy, U.S.R. Murty, Graph Theory with Applications, Macmillan Press, 1976.
[8] G.A. Dirac, Some theorems on abstract graphs, Proc. London Math. Soc. (3) 2 (1952) 69–81.
[9] T. Feder, R. Motwani, C.S. Subi, Finding long paths and cycles in sparse Hamiltonian graphs, in: Proceedings 32th STOC, 2000, pp. 524–529.
[10] E. Flandrin, H.A. Jung, H. Li, Degree sum, neighbourhood intersections and Hamiltonism, Discrete Math. 90 (1991) 41–52.
[11] W. Kocay, P.C. Li, An algorithm for finding a long path in a graph, Util. Math. 45 (1994) 169–185.
[12] B. Korte, J. Vygen, Combinatorial Optimization: Theory and Algorithms, Springer-Verlag, Berlin, 2000.
[13] D. Karger, R. Motwani, G.D.S. Ramkumar, On approximating the longest path in a graph, Algorithmica 18 (1997) 82–98.
[14] B. Monien, How to find long paths efficiently, Ann. Discrete Math. (1985) 239–254.
[15] O. Ore, Note on Hamilton circuits, Amer. Math. Monthly 67 (1960) 55.
[16] S. Vishwanathan, An approximation algorithm for finding a long path in Hamiltonian graphs, in: Proceedings 11th SODA, 2000, pp. 680–685.
[17] B. Wei, Longest cycles in 3-connected graphs, Discrete Math. 170 (1–3) (1997) 195–201.